2018

Measuring Goal Similarity Using Concept, Context and Task Features

Vahid Eyorokon
Wright State University

MEASURING GOAL SIMILARITY USING CONCEPT, CONTEXT AND TASK FEATURES

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science

By

VAHID EYOROKON
B.A., Wright State University, 2016

2018
Wright State University

WRIGHT STATE UNIVERSITY

GRADUATE SCHOOL

July 26, 2018

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY Vahid Eyorokon ENTITLED Measuring Goal Similarity Using Concept, Context and Task Features BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Master of Science.

Michelle A. Cheatham, Ph.D.
Thesis Director

Mateen M. Rizki, Ph.D.
Chair, Department of
Computer Science and Engineering

Committee on
Final Examination

Michelle A. Cheatham, Ph.D.

Michael T. Cox, Ph.D.

Michael L. Raymer, Ph.D.

Barry Milligan, Ph.D.
Interim Dean of the Graduate School

ABSTRACT

Eyorokon, Vahid. M.S., Department of Computer Science and Engineering, Wright State University, 2018. Measuring Goal Similarity Using Concept, Context and Task Features.

Goals can be described as the user's desired state of the agent and the world and are satisfied when the agent and the world are altered in such a way that the present state matches the desired state. For physical agents, they must act in the world to alter it in a series of individual atomic actions. Traditionally, agents use planning to create a chain of actions each of which altering the current world state and yielding a new one until the final action yields the desired goal state. Once this goal state has been achieved, the goal is said to have been satisfied. Since these goals involve physical actions, we can describe these goals as being physical goals. Our work focuses on a special type of goal that doesn't exist physically and are knowledge goals. Much like physical goals, knowledge goals also have a desired state but this desired state is of the user's understanding. Once the user has learned the missing information, the knowledge goal has been satisfied. While physical goals are given to agents who must then produce a plan of actions to alter the world, knowledge goals are given to an agent who must then produce a sequence of intermediate knowledge goals to alter the user's state of knowledge. Much like how individual actions comprise a plan to alter the physical world, individual questions comprise a goal trajectory and alter the state of a user's knowledge. This overall path of inquiry is much like that of an investigation for knowledge not unlike those of a detective or investigator. Given that not all users learn the same way, creating a plan to solve a knowledge goal is not a trivial task. Furthermore, in complex domains, it is not immediately clear to user themselves what their knowledge goal is as they continue to understand how to phrase the correct questions. As

the user continues to refine their questions, their search grows in length and often in complexity as questions become increasingly specific. To address these issues, we created and evaluated a case-based goal reasoning system with the ability to measure similarity between goals.

Table of Contents

List of Figures

List of Tables

Acknowledgements

This work would not have been possible without my thesis advisor, Dr. Michael Cox. His passion for advancing the field of artificial intelligence and his work ethic are truly compelling and something I hope to one day achieve. I would also like to thank Dr. Michael Raymer and Dr. Michelle Cheatham for their advice and helpful feedback.

I would also like to thank Srikanth Nadella for pointing me to related work done by other researchers. To my lab mates, thank you for assisting with evaluations, for listening and providing suggestions. A special thanks to all faculty and professors at Wright State University in both the Computer Science and Philosophy departments for instilling in me a deep appreciation for knowledge, logic and the scientific method.

I owe a great debt of gratitude to my family: Mojdeh, Vincent, Natasha and Parrissa Eyorokon; to my incredible extended family and friends. Thank you all for your unconditional solicitude.

This material is based in part on research sponsored by the Air Force Research Laboratory, under agreement number FA8650-16-C-6763 and the Air Force Office of Scientific Research under grant number FA2386-17-1-4063. This research was also supported by ONR grant N00014-15-C-0077. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the

authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory, the Air Force Office of Scientific Research, the Office of Naval Research or the U.S. Government.

1. Introduction

1.1. Overview

People use information systems to gather information and answer questions they might have. In some sense, this can be thought of as an investigation similar to one a detective might perform with the objective of acquiring knowledge. If a user lacks the ability, skill or experience to effectively initiate and subsequently complete an investigation independently, one can argue that such a user would benefit from guidance. Without the necessary background, the user may feel it is difficult to even phrase the right questions as they lack the ability to succinctly describe their problem. Furthermore, the resources that a user has at their disposal can affect stress levels which can impact their performance. Over time, their investigation may become increasingly complex as they refine their understanding. This dynamic environment can be difficult to navigate for both the user and system as the types of questions the user asks change in direction and level of sophistication.

All of these issues together create a challenge for any system that is designed to assist users through evolving investigative processes. In order to effectively guide the user, a system must be implemented in an intuitive way that does not add stress, while being able to track the user's progress. We assert that a system with the ability to reason about the user's goals can address these issues with a core component being its ability to measure the similarity between goals.

When a user has a need for more information, they often express this need in the form of a question. Question and answer systems like StackOverflow or Quora are powerful tools when it comes to assisting users in knowledge tasks. For simple domains, these systems easily satisfy the user's requirements when they need answers to an isolated and individual query. By retrieving the answer to the question, these systems are effective with simple knowledge tasks.

However, in more complex domains where investigations consist of multiple questions in a series, as in dialogues, these basic systems are unable to consider past or other relevant information (Eyorokon et. al, 2016). Without an effective method for measuring goal similarity, the system's ability to assist the user is bounded which can lead to confusion, mistakes and prolonged knowledge investigations. Depending on the domain investigations may bear some resemblance to one another. For this reason, we assert that a case-based reasoning system with the ability to reason about goals would be suitable for knowledge investigations.

For a case-based reasoning system to be effective, it must first have the ability to adequately represent the user's goals as they arise in an investigation (Bengfort and Cox, 2015) as well as the ability to measure the similarity between goals. We evaluate methods for measuring goal similarity and its application in core functions of case-based reasoning systems like retrieval and reuse of cases. Goal similarity can then be used to identify not just best matching questions, but entire dialogues useful for domains with complex knowledge investigations. Goal similarity can also be used to help the user avoid making mistakes and affords the system with the ability to identify where a user made one.

1.2. Current Research

Search for information on complex issues and topics is challenging as well as ubiquitous in a modern society and the knowledge economy. Much like a criminal investigation, a knowledge investigation revolves around a series of key questions or knowledge goals that seek to provide answers related to a central purpose of the investigation (Bengfort and Cox, 2015; Ram and Hunter, 1992). The field of this research is in case-based reasoning which is a subset of artificial intelligence and more specifically, machine learning and is also related to goal reasoning and can benefit systems that utilize goal reasoning, like the meta-cognitive dual cycle architecture (MIDCA) (Cox et. al, 2016).

Case-based reasoning leverages a system's past experience of problem/solution pairs called cases which are stored in a case-base which functions like a database. Measuring goal similarity allows a system to better assist the user by comparing their current knowledge goals with any previous user's knowledge goals. Once goal similarity has been measured, the system can then identify faster solutions to satisfying the current user's goal by analyzing the previous user's solution. Approaches that leverage prior experience are very suitable in cased-based reasoning systems, for this reason we developed one such system called Ronin.

1.3. Contribution

This thesis work makes the following contributions:

1) Implements a solution to the problem of representing knowledge investigations as trajectories within a case-base;

2) Implements a solution to the problem of representing knowledge-investigations in a case-base;

3) Uses goal similarity when retrieving knowledge-investigations from a case-base;

4) Uses goal similarity to address the problem of identifying mistakes in the user's dialogue;

A representation of knowledge goals must be simple enough that it can be precisely defined in terms of data structures while rich enough in information for goal reasoning to be effective. For this work, we used knowledge goals from two complex domains which are a concierge domain and a military domain. In the concierge domain, Ronin (our system) took the role of a hotel concierge who answered questions from various hotel guests as they inquire about nearby entertainment, food, activities, safety and other assorted tourism related themes. In the military domain, Ronin assisted analysts with answering intelligence questions.

1.4. Outline of Thesis

The topics for this thesis will be presented in the following format. Chapter 2 provides an overview of case-based reasoning, their applications and the system we developed. Chapter 3 describes knowledge goals and how they are used in case representations for goal trajectories. Chapter 4 covers the four textual similarity measures used for case retrieval and case reuse which are covered in Chapters 5 and 7. Chapter 6 reviews the evaluation for case retrieval. Tangent Recognition and Anomaly Pruning is outlined in Chapter 8 with the evaluation being presented in Chapter 9. Chapters 10 and 11 cover related research and a discussion respectively.

2. Case-Based Reasoning and Ronin

Case-based reasoning (CBR) (Aamodt & Plaza, 1994; de Mantaras et. al, 2005; Kolodner, 1993; Leake, 1996; Riesbeck ,1989) is an approach to problem solving that reuses experience (i.e., cases) rather than solving problems repeatedly from scratch. These systems are effective where problems have similar structures thereby allowing CBR systems to leverage their experience. As described by de Mantaras et. al (2005), these systems perform four basic functions: retention; retrieval; revision and reuse. *Retention* is the ability for the system to save cases from new interactions with users. *Retrieval* is the process of finding the best matching case from the case-base where the problem is closely related to the current user's problem. *Revision* is the process by which the system modifies the retrieved case's solution. This revised solution is then reused by the current user to solve their current problem. *Reuse* is the process where the case acts like a template and is used to help the user find a solution to their current problem. For our work on knowledge investigations, cases are reused to help users avoid making mistakes (Eyorokon et. al, 2018).

2.1. Research Focus

Our work focuses mainly on two of the CBR phases however, our case representation as described in Chapter 3 is relevant to case retention.

- **Case Retrieval**: Compares the user's current dialogue in an investigation along with information about the user and their task with each case in the case-base. A scalar value is then calculated for each dialogue as it relates to the current user's dialogue. Cases are sorted accordingly and returned in a set of cases called a retrieval set.

- **Case Reuse**: After a case has been retrieved, it can be used to guide the user in their current dialogue. Using a case dialogue as a template, the system can keep a user on topic. Should the user ask irrelevant questions, the system can detect these questions which we call tangents.

Each time a user concludes a novel interaction, the system uses the information collected from that interaction to form a data structure to preserve the problem the user had and the solution they found. This problem-solution pair comprises a case and is stored by the system in a case-base which functions similarly to a database. When a new user begins an interaction, the system compares the new user's problem with those of previous cases from the case-base. In *conversational CBR systems* (Aha et. al, 1999; Branting et. al, 2004), user interactions are used to incrementally build a dialogue through iterative process. These dialogues then form cases which are stored in a case-base.

Textual CBR (Recio et. al, 2007; Weber et. al, 1998; Weber et. al, 2005) systems share fundamental problems found in natural language and text processing. Users interact with Ronin by posing a series of questions each of which represents the utterance of a knowledge-goal (Eyorokon et. al, 2017). By posing these questions, or knowledge goals in a series, users incrementally create dialogues (Gu & Aamodt, 2006) that preserve the order in which individual knowledge goals were asked.

2.2. Ronin

Our proposed approach for complex knowledge goal reasoning is a case-based reasoning system called *Ronin*[1] (Eyorokon et. al. 2016) which reuses past experience in an interactive fashion. Ronin is a part of a larger knowledge management system called *SAMURAI* (*Situational Awareness via Mixed-initiative Universal Recognition, Analysis, and Inference*) (Bengfort and Cox 2015). Interactive CBR operates similarly to conversational case-based reasoning systems, which incrementally elicit a target problem through an interactive dialog with the user, attempting to minimize the number of questions before a solution is reached (Aha 2005). To provide an adaptable, investigative system, the methodology we explored guides the user in a finite length interactive dialogue, removing the requirement to minimize session length to facilitate an ongoing discovery process. Additionally, the system itself is a learning agent with the goal of predicting future knowledge goals and acquiring the information in advance to provide specific guidance to the user. Fig. 1 shows Ronin's dialogue styled interface.

[1] A rōnin was a samurai with no lord or master during the feudal period (1185–1868) of Japan.

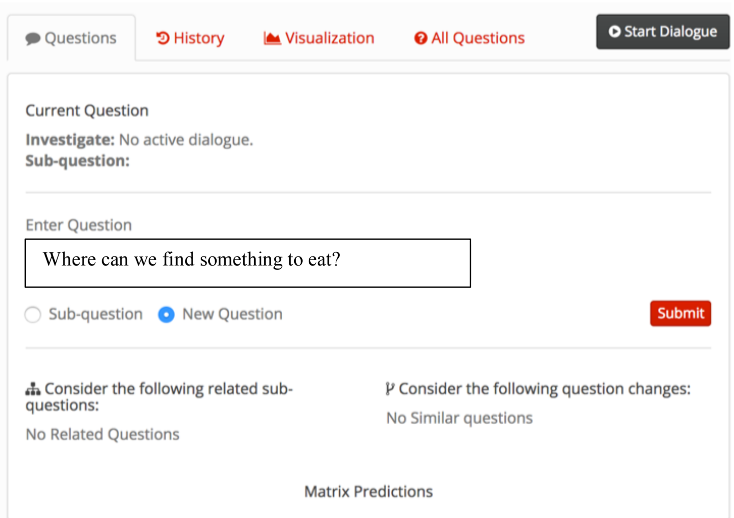

Figure 1. Ronin's dialogue styled interface.

In such systems, one type of mistake manifest as off-topic questions. We consider these misguided knowledge goals to be *tangents*. These appear as anomalies in a hyper-dimensional similarity space generated from a dialogue's comprising questions using a technique we will discuss. We call this goal reasoning process *tangent recognition*. Once detected, anomalies can then be pruned or removed via a separate process we call *anomaly pruning*. Fig. 2 shows the visualization of tangents in a similarity space within Ronin's interface.

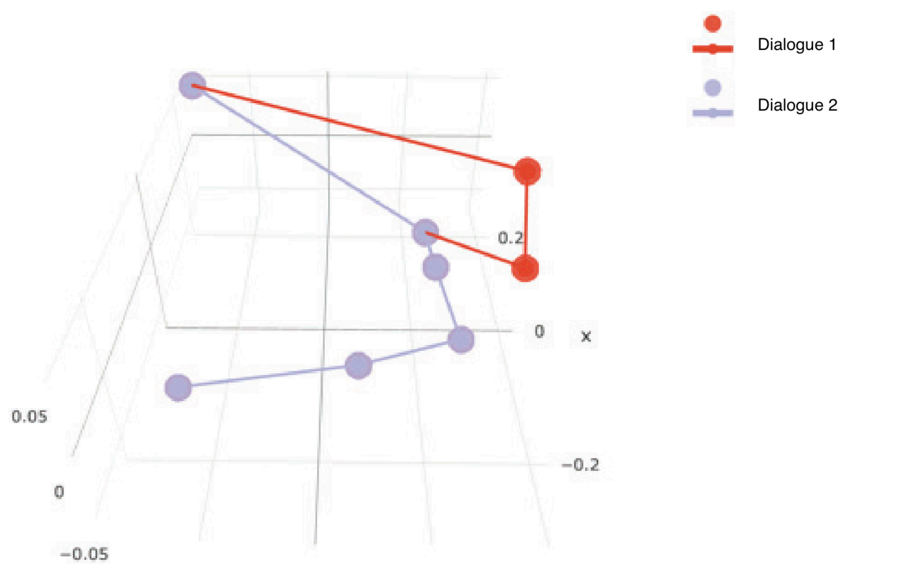

Figure 2. Two dialogues represented as goal trajectories in a similarity space where each node shows the position of a question. Both dialogues share the same questions, except dialogue 1 contains tangents shown in red.

By developing an algorithm to recognize these tangents as they occur, a system can better assist its user in a knowledge investigation and help users avoid tangents in real-time. This creates an opportunity to save the user time on an otherwise protracted search for new information. Additionally, it affords a system with the ability to identify precisely where a user made a mistake and a chance to give the user immediate feedback. We evaluate an algorithm for tangent recognition combined with anomaly pruning that enables a system to trap unrelated questions as a user asks them.

3. Case Representation: Knowledge Goals and Goal Trajectories

Traditionally, agents use planning to create a chain of actions for the purpose of altering the current world state and yielding a new one until the final action yields a final, desired goal state. When this state has been reached, the goal is said to have been satisfied. Because these goals typically involve physical actions, we can describe such goals as being physical goals. Our work differs in that we focus on a special type of goal that doesn't exist physically. We call these goals, knowledge goals. Much like physical goals, knowledge goals also have a desired state, but this desired state is of the user's understanding. Once the user has learned the otherwise missing or unknown information, the knowledge goal has been satisfied. While physical goals are given to agents who must then produce a plan of actions to alter the world, knowledge goals are given to an agent who must then produce a sequence of intermediate knowledge goals to alter the user's state of knowledge. Just as individual actions comprise a plan to alter the physical world, individual questions comprise a goal trajectory and alter the state of a user's knowledge. This overall path of inquiry is not unlike investigations a detective might follow.

A knowledge goal is not the same as the utterance of a question. A *knowledge goal* is the needed information or knowledge that would satisfy the user's desired state of knowledge (Ram & Hunter, 1992; Ram, 1991). By acquiring this information, the user's state of knowledge transitions to a new state where they have learned the previously missing information and the user's knowledge goal is said to have been satisfied. Knowledge goals often can be expressed in the form of the utterance of a question, where the utterance of the question is the most superficial part of the entire knowledge goal, yet we may sometimes use knowledge goals and questions interchangeably.

3.1. Knowledge Goals

Knowledge goals appear in various domains and can be decomposed into three simpler components of concept, context and task. One such domain is in cyber security where information is constantly changing due to evolving threats (Panjala et. al, 2017). In this work we considered two domains which often have complex knowledge investigations: a military domain and a concierge domain. Within the concierge domain, our CBR system took the role of the concierge at a hotel desk to answer questions from guests. In the military domain, our CBR system assisted users when developing strategies within dynamic environments. These strategies involved patterns of questions and answers, much like dialogues in the concierge domain.

However, before our system can perform any CBR related functions (retrieval, revision, reuse, retention) using these natural language utterances, our system must extract useful information from them. First, by understanding concepts and ideas that appear in the user's knowledge goals. Next, we identify important background or contextual information about the user that may be relevant. Finally, we determine the underlying reason or task that is the user's motive for asking the knowledge goal when initiating a dialogue. Thus, these components of: concept, context and task; represent the three components of a knowledge goal (Bengfort & Cox, 2015).

3.2. Goal Trajectories

Our system uses a dialogue-styled interface where the user can ask a series of questions. This interface produces a data structure that is a chain of knowledge goals which we call a *goal trajectory* (Eyorokon et. al, 2016). The structure of the chain preserves the order in which knowledge goals have been asked. Goal trajectories thus have a beginning and an end, and we can consider the evolution of ideas by moving in that direction. When the dialogue begins, the user is asked to enter a preface or task, for the dialogue. Information about the context can be captured by the user's profile. As knowledge goals are posed to our system, each goal's utterance contributes additional conceptual information as the questions reference new ideas.

By chaining together these knowledge goals as shown in Fig. 3, we create a goal trajectory. This text-based goal trajectory then becomes a case in our case-base. The key difference between a dialogue and a goal trajectory is that a goal trajectory is the case representation of knowledge goals in series within a similarity space, but the two terms may sometimes be used interchangeably.

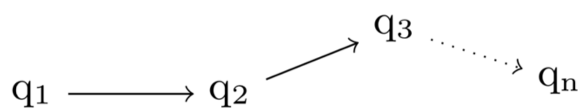

Figure 3 A case as a knowledge goal trajectory.

During this process, the search plan can change as the user discovers new information and forms new questions; indeed, the questions themselves can change. Like attainment goals (i.e., goals to achieve world states), that are subject to transformation (Cox et. al, 2017; Cox & Veloso 1998), we claim that a knowledge goal is also subject to change. Therefore, as interactive reasoning changes a knowledge goal, the path that leads to the final information can be represented by a goal trajectory.

During the dialogue, the system tracks goal changes by recording when new questions are posed. The dialogue is completed by a final knowledge-goal, presumably the target of the investigation with the assumption that user did not give up or abandon their investigation. A knowledge investigation I, is represented with an initial goal g, user u, start time t_1,[2] end time t_n, length n and a Boolean (= ⊤ if successful). Here, *name* is a string from the alphabet Σ. We use *d(I)* to refer to the function that returns the dialogue from *I*.

$$I = (d, u, t_1, t_n, n, succ) : D \times User \times \mathbb{N} \times \mathbb{N} \times \mathbb{N} \times \{\bot, \top\}$$

$$d[1 \ldots n] = q_1 \mid d[2 \ldots n] = \langle q_1,\ q_2 \ldots\ q_n \rangle$$

[2] Time is a positive integer representing the number of seconds elapsed since the UNIX epoch (i.e., January 1, 1970).

$$User = \{(name, age, gender, mstatus) \in \Sigma^* \times \mathbb{N} \times \{male, female\} \times \{single, married, divr, widow\}\}$$

$$C = \{(d, u, t_1, t_n, n, succ) \in D \times User \times \mathbb{N} \times \mathbb{N} \times \mathbb{N} \times \{\bot, \top\}\}$$

3.3. Knowledge Goal Features

While question and knowledge goal may be used interchangeably, a question is not the same as a knowledge goal but is instead the most superficial part of the knowledge goal itself. A knowledge goal is the need for more information which can be expressed as a question. A question itself may capture the ideas the user has in mind, but the question's utterance cannot be the only thing used to answer a question since questions can appear in very different contexts, for very different reasons. For this reason, we can see how identical utterances of questions can have very different answers. The objective of this work is not to address the issue of answering questions. We seek to build on the work done by (Bengfort and Cox, 2015) of representing knowledge goals to address the challenges of representing knowledge goal investigations and how goal similarity can be used to reason about those investigations to help future users.

To satisfy a knowledge goal, it is important to consider the concepts of the question as well as the context of who is asking it and the task that they need to fulfill. Thus, concept, context and task were chosen as features used to represent knowledge goals. Consider this seemingly trivial utterance of a knowledge goal from the concierge domain: Where can we find something to eat? When asked by honeymooners at noon, an appropriate response might suggest something romantic like a sit-down restaurant or even a picnic on the beach. However, a completely identical utterance can also be expressed by two bachelors at 2:00

a.m. For the concierge to suggest a picnic at the beach might lead to confusion which would delay the bachelors by prolonging their knowledge investigation. For this reason, knowledge goals should be represented as concept, context and task and their utterance q is part of Q, or the set of all possible questions which are strings from the alphabet Σ^*.

$$Q = \{(c, u, \tau) \in Concept, Context, Task\}$$

$$q \in Q$$

3.3.1. Concept

The ideas that appear within the utterance of a knowledge goal represent the concept. When considering our example: Where can we find something to eat?, the word "eat" is part of the verb phrase and would be parsed by our CBR system which uses the Stanford Parser (Schuster & Manning, 2016) This term is a concept that gives our CBR system a clue about the ideas which should appear in the answer as well as when identifying related knowledge investigations. Terms related to eating and food should be given a higher priority.

3.3.2. Context

As part of our CBR system's interactive process, users are required to create profile accounts to better understand the context of a question (Hwang et. al, 2012). These accounts include information regarding their: gender, marital status, sex, location, biography, etc. Attributes that describe characteristics and traits of the user are part of the context and for this were they were arbitrarily chosen. Users can create groups based on interests and join them. They can also be assigned to teams based on their skill-set and experience. Our system can then leverage this contextual information to compare user profiles and measure the similarity of individual users themselves.

3.3.3. Task

The task is the need or motivation for initiating a knowledge investigation (Ram & Hunter, 1992) and often appears as the preface to a dialogue. In the concierge domain, guests at a hotel will approach the concierge and ask a series of questions. Usually, before the first question is even asked, guests will give a preface that outlines the reason for the questions they are about to ask. If instead guests approached the concierge and promptly began firing away a series of questions, this would be perceived as robotic and socially awkward. For this reason, generally dialogues are preceded by a preface, or task.

In the concierge domain, the guest would first greet the concierge at the desk. Then, the guest would preface their dialogue with their task and state something like: Our plane was delayed for 6 hours and we finally arrived. This would then be followed by the first question: Where can we find something to eat? With this information, the concierge would realize that time might be of a higher priority and therefore fast food might be a better response than suggesting a sit-down restaurant that requires a reservation. From here, the dialogue will naturally continue as a series of questions. Yet if we hadn't considered task, how else would we have known simply from the individual question that time was important for the user?

Often, work with question/answering systems do not consider task. Yet the task we've described clearly provides crucial and relevant information. As the dialogue progresses, clarifying details may be introduced as new tasks and knowledge investigations become increasingly complex. Representing dialogues is crucial for the success of a case-based reasoning system. We outline a case representation for dialogues using a goal trajectory.

4. Text Vectorization and Goal Similarity

Our system can use four different ways of measuring similarity between knowledge goals. Since each goal's concept can be represented as the utterance of a question, we can exploit this utterance using existing textual similarity measures when determining the similarity between questions. Such similarity measures are also used in case trajectory retrieval as described in (Eyorokon et. al, 2018). The first of these is Term Frequency Inverse Document Frequency which is a statistical, bag-of-words approach (Harris, 1954). The second is Word2Vec which is a neural network that generates vectors for a word by considering its surrounding words. These vector representations can be compared to measure the similarity between words and yield a similarity score when using cosine similarity. We will discuss how word vectors can be used to measure similarity between sentences. The third measure uses a semantic net, along with corpus statistics and is an algorithm based on the work done by (Li et. al, 2013) which we refer to as NetSim. The fourth is called SkipThoughts and uses a neural network similar to Wod2Vec. We will discuss each similarity measure, their advantages and their drawbacks.

4.1. Term Frequency Inverse Document Frequency

Term Frequency Inverse Document Frequency (TFIDF) requires a rich corpus of text for the bag-of-words approach to be effective. For this, our system relies on parsing the noun and verb phrases from each utterance and querying the phrases to Wikipedia. This query usually returns a related page from which our system extracts the first three sentences. These sentences are added to the original text of the question's utterance to build a question document. This process is done for each utterance in the database and TFIDF is performed using the resulting question documents instead of the utterances alone and allows us to gain a better understanding of conceptual terms in the question (Huang et. al, 2009). However, this process of querying Wikipedia introduces the issue of disambiguation. Sometimes, Wikipedia will return a list of possible results that are related to the queried phrase instead of a single page. To disambiguate these results would require a sophisticated algorithm or constant human supervision. Given the challenges of disambiguation and since our research focuses on retrieval, we default to the result Wikipedia suggests is the best matching. Yet this is sometimes incorrect.

4.2. Word2Vec

Word2Vec (Mikolov et. al, 2013) is a neural network trained on a large corpus of textual data collected by Google. This approach creates a vector representation for words that considers surrounding words. When an utterance is provided, each word in the string is converted into a vector (Salton et. al, 1975). After gathering the vector representations for each word in the utterance, we averaged these vectors into a single vector for the entire utterance. That is, given a vector for each individual word in a sentence we compute the sum of all vectors divided by the number of words in the sentence to form a sentence vector (Singhal, 2001). The main insight in this method is to obtain the individual word embedding vectors in a question/sentence and form a sentence embedding by averaging vectors. By using the cosine similarity (Kryszkiewicz, 2014) between two questions vectors we calculate the similarity. We obtain each word embedding from the Word2Vec skip-gram model which was pre-trained on Google News vectors containing a corpus of 3 billion words. Word2Vec's skip-gram algorithm predicts whether a word belongs to the surrounding window of words, from a three-layer neural network with one hidden layer while both input and output layers being the unique bag of words thereby forming a word embedding.

4.3. SkipThoughts

The *Skip-thoughts* model draws inspiration from the skip-gram structure in the word2vec model. It consists of a neural network that is trained to reconstruct the surrounding sentences that share syntactic and semantic properties (Kiros et. al, 2015). This model is based on an encoder-decoder architecture where encoder maps natural language sentences into fixed length vector representations. Then given the vector representation of a sentence, the encoder is built using recurrent neural network layers, bi-directional recurrent neural network layers, or a combination of both. This captures the temporal patterns of continuous word vectors. Hidden states of the encoder are fed as a representation into two separate decoders. Again, each decoder uses another set of recurrent layers. These decoders share the same look-up table with the encoder and then predict the preceding and subsequent sentences.

4.4. NetSim

NetSim is based on an algorithm described by (Li et. al, 20016) which uses a semantic net along with corpus statistics to measure the similarity between two sentences. The semantic net factors into account two measures: semantic and syntactic similarity. Semantic similarity looks at the synonyms words have in common, the distance from one word to another in the semantic net and the depth of a word in the semantic net. Since depth of a word relates to the specialization of a word, distance alone cannot be used. To understand why depth is important, consider the following example. The word 'human' may appear closer to the word 'boy' than the word 'babysitter' but a knowledge goal about humans may be less relevant than one about 'babysitters'. Since words become more specialized as we go down the semantic net, depth is factored into the similarity measure. Syntactic similarity for NetSim considers the position of words in one utterance and the distance from related words in another utterance.

Additionally, *inverse document frequency* (IDF) is used to establish an information content of each word. This is a statistical measure where words that are common in the corpus have a low information content and thus a lower IDF score, while less occurring words have a higher IDF score. Finally, the semantic and syntactic scores are multiplied by weights and added together yielding a scalar value for similarity. The semantic and syntactic weights for our evaluation were set to 70% and 30% respectively. It should be noted that the use of a semantic net has drawbacks since they only capture is-a relationships and while similarity using NetSim is powerful, it is also bounded as we will see from the evaluations.

5. Case Retrieval

Retrieval is a key function for a case-based reasoning system. These systems store past cases and their solutions in a data structure called a case. By identifying related cases, the case-based reasoning system can retrieve the best matching case so that the solution can then be adapted and used as a solution to the current problem. For our conversational case-based reasoning system, cases are entire goal trajectories. Retrieving the best matching goal trajectory requires an algorithm that considers the contextual, conceptual and task similarities of each goal trajectory's comprising knowledge goals. We begin by determining the conceptual similarity.

5.1. Concept Similarity

For a given knowledge investigation, $I = (d, u, t_1, t_n, n, succ)$, our system performs retrieval by considering the similarity between questions in the user's current dialogue d_c against questions that appear in dialogues from the case-base $I_1 - I_m$ as shown in Fig. 4. By calculating pair wise similarity between the utterance of questions, we iteratively consider each index in d_c against the corresponding question at the same index in each case d_1 - d_n.

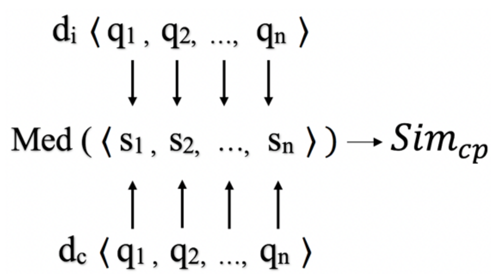

Figure 4. The process for measuring concept similarity Sim_{cp}.

This process produces a *vector of similarity scores* Sim_{cp} for each pair of dialogues, whose length is equal to the shorter length of either d_c or a candidate dialogue from the case-base *C*. From this vector of similarity scores, the median value is yielded as the concept similarity which we refer to as Sim_{cp}. The next step is to capture contextual similarity.

5.2. Context Similarity

Since every dialogue in the case-base has a user, our system can compare the user profiles from the current user *x* against that from a dialogue in the case-base denoted as *y*. User profiles are a rich source of attributes that include their age, marital status, gender etc. Specific traits we used were arbitrary and rather we meant to highlight our retrieval algorithm's flexibility when considering contextual information relevant to the problem domain. For scaling features like age, a threshold is set. Fig. 5 shows the context similarity function.

$$Sim_{cx} = \frac{\sum_{1}^{n} f(x,y)}{n}$$

Figure 5. A function for measuring context similarity between two user profiles denoted as *x* and *y*. By iterating through all n user profile features, the number of matching features is recorded and then divided by *n*.

In our system, age had a threshold of four years and any difference between the two user profiles under the threshold was considered a match. For binary and enumerable features, matching was straightforward and trivial. Given our representation, each user profile can also be thought of as a set of attributes. Attributes were considered on a matching basis where pairs of attributes from the current user and a case user were passed to a matching function. This function returned a one or zero if the attributes were matched. The sum of total matches is divided by the total number of attributes referred to as *n* to yield a final contextual score we refer to as Sim_{cx}.

5.3. Task Similarity

When a user begins a dialogue, our system asks them to provide task information in the form of a single sentence preface. By using one of the four similarity measures for text, we can extract a score for task similarity between the current trajectory d_1 and a case trajectory d_c which we refer to as Sim_{tk} Finally, we calculate the overall goal trajectory similarity.

5.4. Goal Trajectory Similarity

After each knowledge goal is asked by the user, our system calculates the conceptual, contextual and task similarity to yield a vector (Sim_{cp}, Sim_{cx}, Sim_{tk}) which represents the similarity between the current user's knowledge investigation and a case trajectory. Having a vector representation for goal similarity allows our system to map the user's goal in a three-dimensional goal space providing visual feedback for the user. Table 1 outlines the retrieval algorithm.

Table 1. Formalization of dialogue retrieval. Refer to Fig. 2 and Fig. 3 for concept and context Similarity.

```
function TASK-SIMILARITY(old-task, current-task)
    Si ← similarity(old-task, current-task)
    return(Si)
function CONCEPT-SIMILARITY(old-case, current-case)
    CpSV ← []
    for i ← 1 to length(current-case) do
        Si ← similarity(old-case[i], current-case[i])
        CpSV.append(Si)
    return(median(CpSV))
function CONTEXT-SIMILARITY(old-case, current-case)
    CxS ← 0
    matches ← 0
    possible ← 1
    if (|old-case.user.age − current-case.user.age| < 4) then
        inc(matches)
    Uc ← current-case.user
    Ui ← old-case.user
    intersection ← (Uc.attributes ∩ Ui.attributes)
    union ← (Uc.attributes ∪ Ui.attributes)
    CxS ← (length(intersection) + matches)/(length(union) + possible)
    return(CxS)
function CASE-BASE-SIMILARITY(case-base, current-case)
    sequence ← []
    for case in case-base do
        CpS ← CONCEPT-SIMILARITY(case, current-case)
        CxS ← CONTEXT-SIMILARITY(case,current-case)
        TkS ← TASK-SIMILARITY(case.task,current-case.task)
        similarity ← average(CpS, CxS, TkS)
        sequence.append((similarity,case.id))
    return(sorted(sequence, key=lambda x: x[0]))
CASE-BASE-SIMILARITY(case-base, current-case)
```

Different domains may have their own need to control the importance of concept, context and task so tunable weights are established for each feature that are then multiplied by their respective score. After the three scores are then weighted, the results are added together to yield a final similarity score between the user's current goal trajectory and a case goal trajectory. These scores are sorted, and goal trajectories are returned in a retrieval set which orders trajectories by most similar to least.

6. Evaluation of Case Retrieval

The first evaluation was performed using dialogues from both the concierge and the military domains. As previously mentioned, in the concierge domain, our system took the role of a hotel concierge who answered questions for hotel guests. In the military domain, our system assisted analysts with answering intelligence questions. We iteratively took each dialogue and paraphrased its questions to semantically equivalent questions where utterances were not exact text matches of the original. Refer to Appendix A. for a full list of cases used. The first line is the task information. Table 2 presents an example paraphrased dialogue.

Table 2. An example dialogue from the case-base and its paraphrased version. The first sentence is the task information.

Original Dialogue	Paraphrased Dialogue
A family is concerned with safety.	A guest wants to know how to stay safe.
How safe is it at night to go out?	Should we be concerned with safety tonight?
Where is the nearest police station?	How can we contact law enforcement?
What are the crime rates in neighborhoods?	Are there any areas that are more dangerous?
What kind of fraud should we know about?	What scams do people use?
How can we contact to police?	What's the number for law enforcement?

Retrieval was performed after each question was submitted and the position of the original dialogue was recorded in the retrieval set. The closer the original dialogue was to the first position in the retrieval set, the better. Two separate evaluations with and without task information will be discussed.

6.1. Retrieval Evaluation Without Task

Here we evaluated the effectiveness of four separate measures: Word2Vec, NetSim TFIDF and SkipThoughts when used in our goal trajectory retrieval algorithm. Each dialogue was at least five questions long and paraphrased questions were entered sequentially up to the first five questions. After each question, retrieval was performed. We refer to the length of the dialogue at the time retrieval was performed as the probe size and appears on the X axis. After retrieval sets were generated for each iteration of our probe size, we averaged the position of the desired dialogue across all retrieval sets for that probe. This averaged value for the position in the retrieval set is shown on the Y axis. The best any similarity measure could do, was an average position in the retrieval set of one.

For this evaluation, no task information was used. In the military domain, 21 dialogues were paraphrased and evaluated at five probes for a total of 105 retrievals each using Word2Vec, NetSim, TFIDF and SkipThoughts for a total of 420 retrievals. The results are shown in Fig. 6.

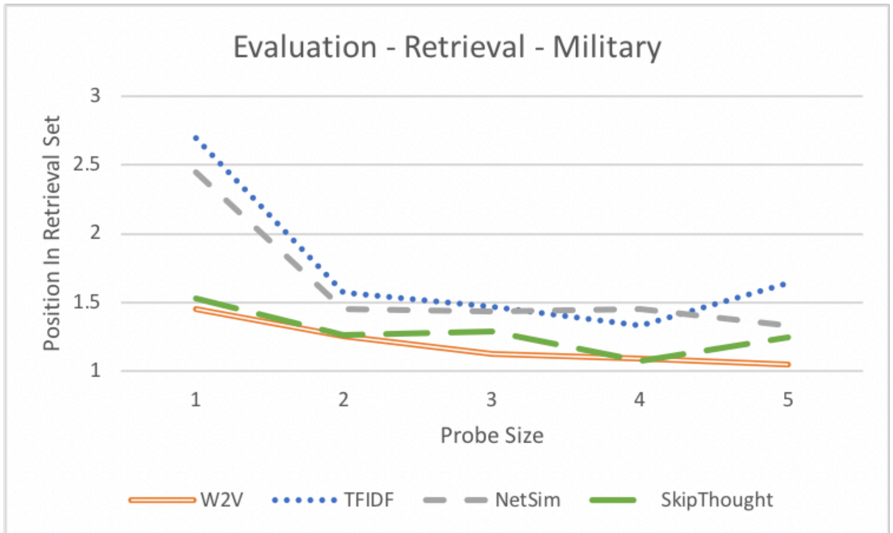

Figure 6. Retrieval evaluation in the military domain.

In the concierge domain 19 dialogues were paraphrased and evaluated at five probes for a total of 95 retrievals each for Word2Vec, NetSim TFIDF and SkipThoughts for a total of 380 retrievals. Overall, retrieval was performed 800 times in both domains. In the military domain, both TFIDF and NetSim performed about the same. TFIDF's performance worsened on probe size five, but aside from this, the performance of all four similarity measures generally improved over time as the position of the desired dialogue approached the first position in the retrieval set. The results are shown in Fig. 7.

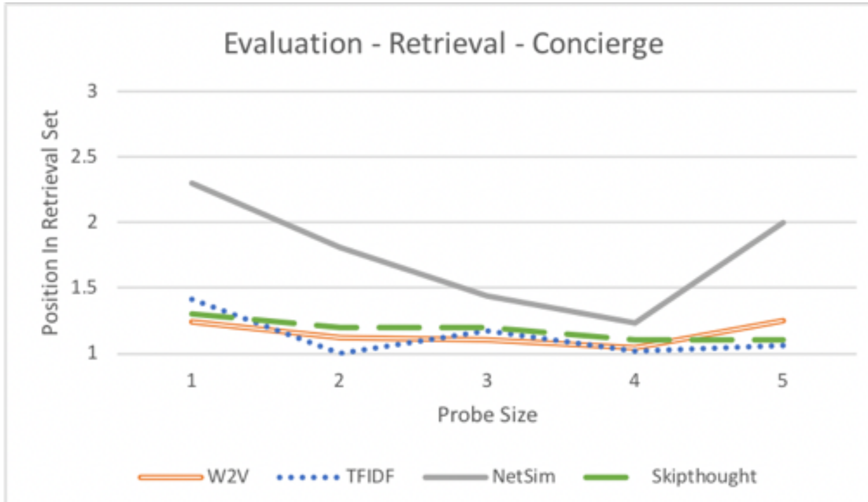

Figure 7. Retrieval evaluation in the concierge domain.

Here we see that our retrieval method when using SkipThoughts and Word2Vec outperformed TFIDF and NetSim in the military domain. In the concierge domain, Word2Vec and TFIDF were closer to the same in performance and both were considerably better than NetSim. As the probe size reached a size of five, NetSim's performance significantly worsened. Similarly, Word2Vec's performance also worsened on probe size five in the concierge domain.

When using Word2Vec our retrieval method consistently returned the correct dialogue in the first position of the retrieval set and was often able to do so on probe size one. Word2Vec also did not have an average position above 1.5 at any probe size in either domain with SkipThoughts having one occurrence above 1.5 in the military domain. When used in our retrieval algorithm, Word2Vec proved to be the most reliable and consistent of the four similarity measures with SkipThoughts following closely.

6.2. Retrieval Evaluation Without Task

The second evaluation included task information. For this evaluation, in addition to paraphrasing the text of each question, task information was also paraphrased for cases. When these paraphrased dialogues were entered in, the position of the original case in the retrieval set was recorded. The results for the military domain are shown in Fig. 8.

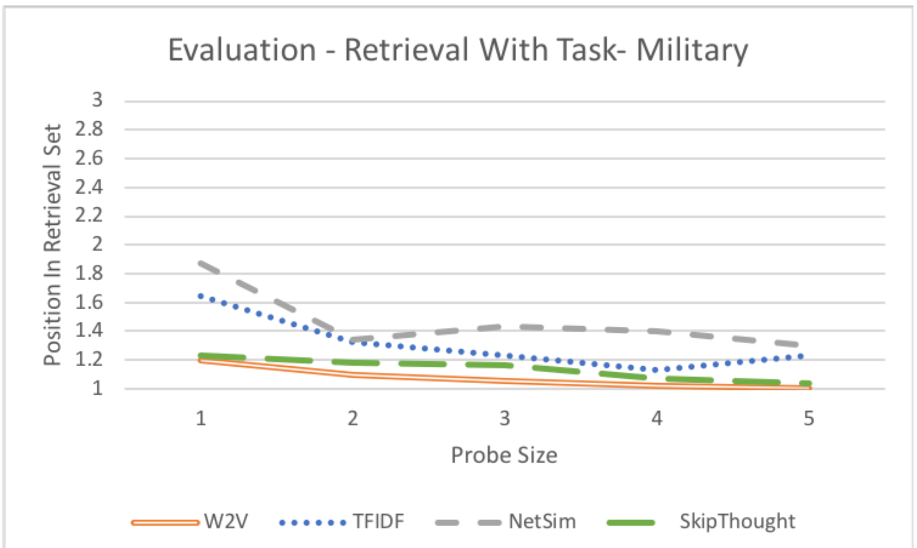

Figure 8. The performance of the case retrieval algorithm with task information. Most generally improved with the exception of TFIDF where task information appeared to add noise.

The addition of task information show improvement in the performance of all similarity measures in the military domain. Retrieval's performance also increased significantly with a length of one for all similarity measures. All similarity measures generally improved as the length of the dialogue increased with the exception of TFIDF which had a slightly worse performance on a length of five.

In the concierge domain, all similarity measures improved in performance overall. With concierge terms being more common, TFIDF's performance in the concierge domain improved as shown in Fig. 9.

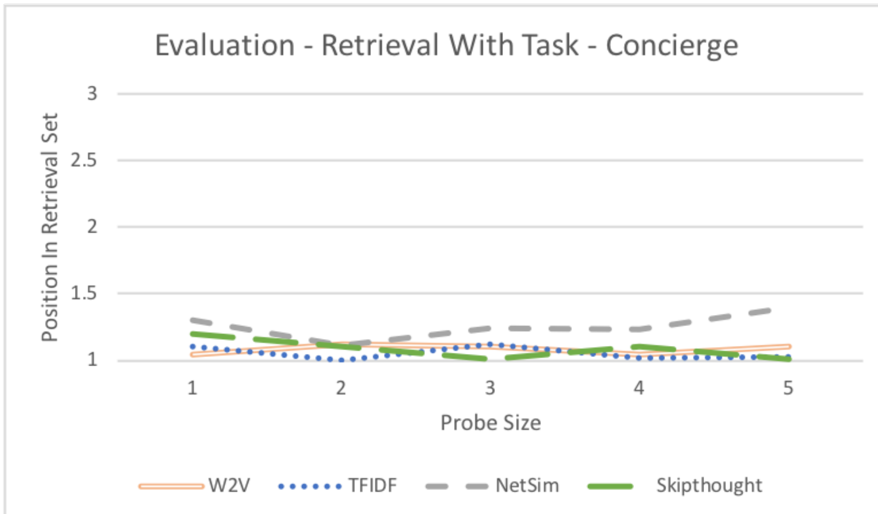

Figure 9. Shows the performance of the retrieval algorithm with the addition of task information.

7. Case Reuse

In any knowledge investigation by which a user must acquire new or missing information, situations often arise which lead to a fork in their investigation. Multiple possible lines of inquiry appear that the users must choose between. A choice of any one would delay the user's ability to choose another if the chosen path proves to be irrelevant and happens to yield only useless information. With limited knowledge or experience, a user must make assumptions which serve as justifications for their choice of a particular path of inquiry. Yet incorrect assumptions can lead the user to choose a path that ultimately leads to dead-end. These fruitless lines of inquiry can waste both time and resources by adding confusion and noise to the user's investigation. Here we evaluate an algorithm called Tangent Recognition Anomaly Pruning to eliminate false starts that arise in interactive dialogues created within our case-based reasoning system called Ronin. Results show that TRAP is an effective algorithm for processing mistakes when reusing cases. Fig. 10. shows how mistakes can manifest in a dialogue.

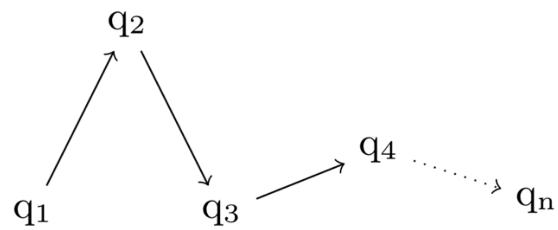

Figure 10. A case where a question (q_2) is off-topic in the dialogue.

7.1. Tangents in Goal Trajectories

When an individual begins their knowledge investigation to acquire new information in response to a problem, it is often the case that their current understanding of the problem itself is ill-defined either through a lack of experience or because the problem domain is challenging. This makes it difficult for the user to phrase questions correctly and accurately enough for their question to sufficiently capture their need for specific information.

Without an accurate question, a system's ability to assist the user is bounded. Additionally, it is not always the case that individuals are afforded the option of having a more experienced and knowledgeable person to guide them through their investigation thereby introducing more exposure to mistakes. Fig. 11 shows a simple representation of recognizing tangents within a dialogue.

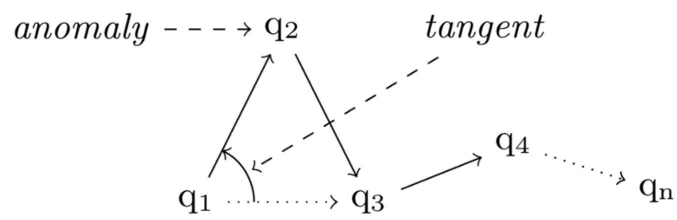

Figure 11. The second question, q_2, is recognized as an anomaly and is shown as a tangent (i.e., the angle between the dotted arrow and the solid arrow).

The medium through which knowledge investigations are performed can be laborious, boring, time consuming or otherwise tiresome as they are often performed through search engines, research journals or basic question and answer systems thereby adding to the stress of the user's investigation. Each scenario can cause mistakes or tangents.

While not all knowledge investigations suffer from all of these properties, we believe it is reasonable to assume that each one increases the chances for mistakes. Fig. 12 shows a tangent being removed from a dialogue.

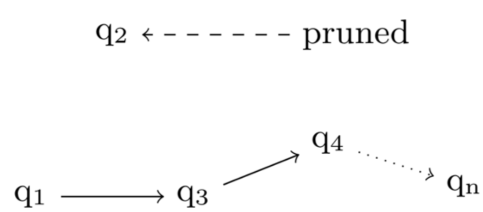

Figure 12. The second question is dropped from the dialogue.

In order to recognize tangents, it is important to consider the types of tangents that exist. Simply put, some tangents are more obvious than others. We use a simple distinction for tangents with two classifications: hard and soft. Our evaluation for this paper will cover hard tangents.

7.2. Soft Tangents

Soft tangents are unrelated questions that are from the same domain. Simply put, these questions are not relevant to the current knowledge investigation, but they are related to the overall domain in which the knowledge investigation exists. Consider an example in the Concierge Domain where a person asks about restaurants nearby. She might ask the concierge for a general list of nearby restaurants. She would then refine this list asking about cost, distance, speed of service and possibly even inquire about specific dietary options. But suppose at some point, the thought of entertainment and museums occurred to her and so she asked about art museums. This question about museums is certainly within the concierge domain, however, it is irrelevant to her current goal of finding a suitable restaurant. For this reason, it is a soft tangent.

7.3. Hard Tangents

Hard tangents are unrelated questions from an unrelated domain. Because hard tangents are from an entirely unrelated domain, intuitively we can surmise that they are easier to detect over soft tangents since soft tangents at least share a similar domain. For hard tangents, we used questions from a political domain. Questions in this domain ask about various global leaders, general practices of democratic systems and laws. When considering our previous example of a person investigating nearby restaurants, a hard tangent would have manifested had she asked about the voting age, the number of senators in the senate or results of the latest general election.

8. Tangent Recognition Anomaly Pruning (TRAP)

We refer to our system's ability to recognize irrelevant questions in the user's dialogue as tangent recognition and the removal of such tangents as anomaly pruning (Eyorokon et. al, 2018). Currently, Ronin finds and removes hard tangents using an algorithm called *Tangent Recognition Anomaly Pruning (TRAP)*. The objective is to eventually develop an algorithm capable of recognizing soft tangents. While hard tangents may be obvious enough that a user can recognize them, they are a good starting point for evaluating TRAP. One area of future research is to recognize soft tangents. The reason why it is important to recognize tangents is because in domains where users have a clear task, as in the military domain, the analyst's ability to satisfy a knowledge investigation may be time sensitive. In such domains, the speed at which an analyst can complete an investigation can mean the difference in lives lost.

The goal of tangent recognition is not merely to represent goal trajectories as a series of Bag-of-Word vectors, but to move beyond this and represent the user's knowledge investigation itself and reason about their goals (Schumacher et. al, 2012). Tangents manifest as anomalies in a hyper-dimensional similarity space and can then be pruned or removed. The process begins by first creating vector representations for questions in a dialogue *d* as shown in Fig. 13.

$$d = \langle q_1, q_2, q_3, \ldots q_n \rangle$$

Figure 13. Dialogue *d* represented as a vector of questions.

Converting each of the dialogue's questions into a vector yields a dialogue matrix that can then be used to recognize tangents and prune anomalies. Each of the following four subsections correspond to the four major functions called by TRAP shown in Table 3.

Table 3. The TRAP algorithm.

Algorithm 1 Tangent Recognition Anomaly Pruning

```
function QuestionVector(origQuestion, goalTrajectory)
    questionVec ← []
    for each question ∈ goalTrajectory do
        sim ← Similarity(question, origQuestion)
        Append(questionVec, sim)
    return questionVec
function DialogueMatrix(goalTrajectory)
    dialogueMatrix ← []
    for each question ∈ goalTrajectory do
        questionVec ← QuestionVector(question, goalTrajectory)
        Append(dialogueMatrix, questionVec)
    return dialogueMatrix
function TangentRecognition(goalTrajectory, threshold)
    dialogueMatrix ← DialogueMatrix(goalTrajectory)
    i, j ← 0, 1
    while j ≤ length(goalTrajectory) do
        pairwiseCosineSim ← cosineSim(goalTrajectory[i], goalTrajectory[j])
        if pairwiseCosineSim < threshold then
            return [i, j]
        increment(i,j)
    return −1
function AnomalyPruning(goalTrajectory, threshold)
    tangentFree ← goalTrajectory
    do
        anomalies ← TangentRecognition(tangentFree,threshold)
        Remove(tangentFree, tangentFree[anomalies[length(anomalies) − 1]])
    while anomalies ≠ −1
    return tangentFree
function TRAP(goalTrajectory, threshold)
    tangentFree ← AnomalyPruning(goalTrajectory,threshold)
    return tangentFree
```

8.1. Question Vector

Each question comprises an atomic part of the overall dialogue. Therefore, the dialogue itself can be represented as a list of its comprising questions. Since the dialogue, or goal trajectory itself can be represented as a list of questions, by comparing a particular original question to each question in the goal trajectory, we can get a sense of the original question's relevance to the overall dialogue. This process can be described in the following: $Sim\ (q, d, f) = f\ (q, d[j]),\ where\ 1\ \leq j\ \leq |d|$. By measuring the similarity of the original question to each question in the goal trajectory denoted as s, we create a question vector that represents that particular question's relevance to the entire goal trajectory. See the QuestionVector function in Table 3 which returns such a representation as that shown in Fig. 14.

$$Sim(q_1, d, f) = \left[\begin{array}{ccccc} s_{q_1,q_1} & s_{q_1,q_2} & s_{q_1,q_3} & \cdots & s_{q_1,q_{|d|}} \end{array} \right]$$

Figure 14. Question q1 represented as a row vector of similarity scores between itself and each question in the dialogue.

8.2. Dialogue Matrix and Similarity Hyper-Space

This question vectorization process is repeated for every question in the goal trajectory and returns in a symmetrical square *n* x *n* matrix where *n* is equal to the number of questions in the goal trajectory. This process can be represented as the following equation: $\mathbb{M}(I)_{1\leq i\leq |d(I)|,*} = Sim(d(I)[i], d(I), f)$. After completing this question vectorization process for all questions in the dialogue, we get a square dialogue matrix where the number of columns and rows are equal to the number of questions in the dialogue. See the DialogueMatrix function in Table 3 which returns the dialogue matrix representation shown in Fig. 15. In such a matrix, each question becomes both a sample and a feature. This matrix represents a multidimensional space of similarity for its comprising questions.

By modeling the user's goal trajectory in a multi-dimensional similarity space, tangents manifest as divergent points along the overall direction of the goal trajectory through the similarity space.

$$\langle\, q_1 \quad q_2 \quad q_3 \quad \cdots \quad q_n \,\rangle \quad \rightarrow \begin{bmatrix} [s_{q_1,q_1} & s_{q_1,q_2} & s_{q_1,q_3} & \cdots & s_{q_1,q_n}] \\ [s_{q_2,q_1} & s_{q_2,q_2} & s_{q_2,q_3} & \cdots & s_{q_2,q_n}] \\ [s_{q_3,q_1} & s_{q_3,q_2} & s_{q_3,q_3} & \cdots & s_{q_3,q_n}] \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ [s_{q_n,q_1} & s_{q_n,q_2} & s_{q_n,q_3} & \cdots & s_{q_n,q_n}] \end{bmatrix}$$

Figure 15. The conversion of each question into a question vector to create a similarity matrix.

8.3. Tangent Recognition

To detect these diverging points, we calculate the pairwise cosine similarity values between adjoining row vectors in the similarity matrix. Cosine similarity yields a scalar value between zero and one corresponding to the similarity of the two vectors in this multi-dimensional space where zero is no similarity and one is perfect similarity. We then compare this value against a tunable threshold. Any values less than this threshold indicate that one of the two vectors is anomalous and represents an off-topic question. See the TangentRecognition function in Table 3.

8.4. Anomaly Pruning

Once a tangent has been recognized, we next need to identify the anomaly. The challenge here is that cosine similarity represents a score between two vectors, but it does not tell us which vector the anomalous or irrelevant question is. Additionally, the length of the pairwise cosine similarity vector is one less than the length of our dialogue. For this we have chosen to make a simple assumption which is one that allowed us to proceed with our evaluation and that we plan to revisit at a later point.

Our assumption is that the first question of the dialogue is never a tangent. Therefore, when given a cosine score that is less than our threshold for determining tangents, the second question of the pair is always selected as the anomaly. See the AnomalyPruning function in Table 3. It should be clear that this assumption lacks consistency in all scenarios as the first question could also be a tangent. For this reason, we plan to explore more robust methods of identifying anomalies like idealization which will be discussed in future work.

After the anomaly has been removed or pruned, the similarity space is reduced accordingly by removing the row and column that corresponded to the anomalous question. After reducing the similarity space, pairwise cosine similarity is recalculated and TRAP repeats until all tangents have been removed.

9. Evaluation of Case Reuse

For this evaluation we used existing dialogues/cases from two separate domains: a concierge and a military domain. While TRAP works in real-time, for our evaluation we used existing dialogues to approximate the effect of a tangent in a dialogue to evaluate the TRAP algorithm. As previously mentioned, in the concierge domain, our system took the role of a hotel concierge who answered questions from various hotel guests as they inquire about nearby entertainment, food, activities, safety and other assorted tourism related themes. In the military domain, our system assisted analysts with answering intelligence questions. We took each dialogue and iteratively inserted hard tangents from a political domain in the middle of the dialogue from one to three consecutive hard tangents. We evaluated TRAP's performance with removing all inserted tangents. The closer the accuracy was to 100%, the better TRAP performed. Conversely, the closer the false positive rate was to 0%, the better.

Here we evaluated the effectiveness of four separate measures: Word2Vec, NetSim, TFIDF and SkipThoughts when used in our TRAP algorithm. Each dialogue in our evaluation was at least five questions long. In the middle of each dialogue, we iteratively inserted one tangent until a total of three were inserted. We refer to the number of tangential questions as the tangent length at the time TRAP was performed and the tangent length appears on the X axis. We performed TRAP using each similarity method and averaged the number of the tangents caught across all dialogues of that iteration's tangent length. This averaged value for the tangent length is shown on the Y axis. We then recorded the number of false positives for that iteration. The best any similarity measure could do, was an average accuracy of 100% and a false positive rate of 0%.

In the military domain, 21 dialogues were used at three separate tangent lengths for a total of 63 TRAP trials each for Word2Vec, NetSim, TFIDF and SkipThoughts. A grand total of 252 TRAP trials were performed in the military domain. In the military domain, TRAP with Word2Vec performed the best with an accuracy of 100% with a tangent length of one as shown in Fig. 16. Similarly when TRAP used SkipThoughts, the performance resembled Word2Vec on tangent length of one, but fell below Word2Vec as the tangent grew. TFIDF performed the second best and NetSim consistently scored the lowest. Most generally worsened as the tangent length increased with TRAP's performance when using NetSim following an arc as it improved on tangent size two.

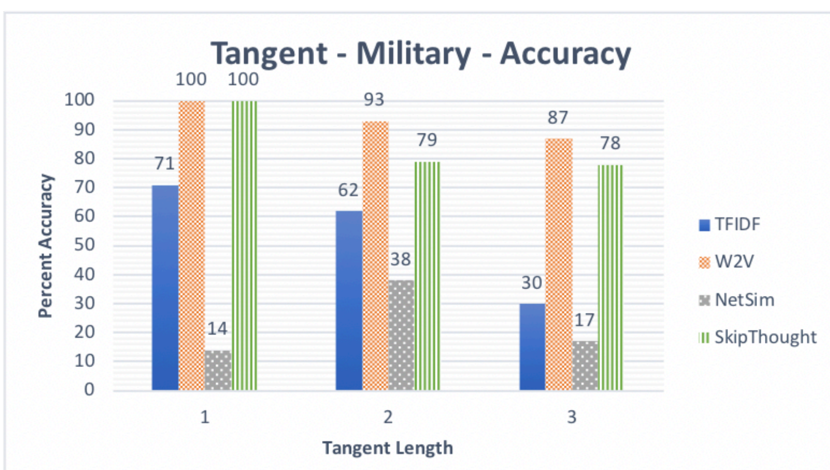

Figure 16. The accuracy of TRAP in the military domain.

All similarity measures generally had similar false positive rates with the exception of TRAP when using SkipThoughts. TRAP with SkipThoughts had a consistently higher false positive rate than all other similarity measures. These results are reflected in Fig. 17.

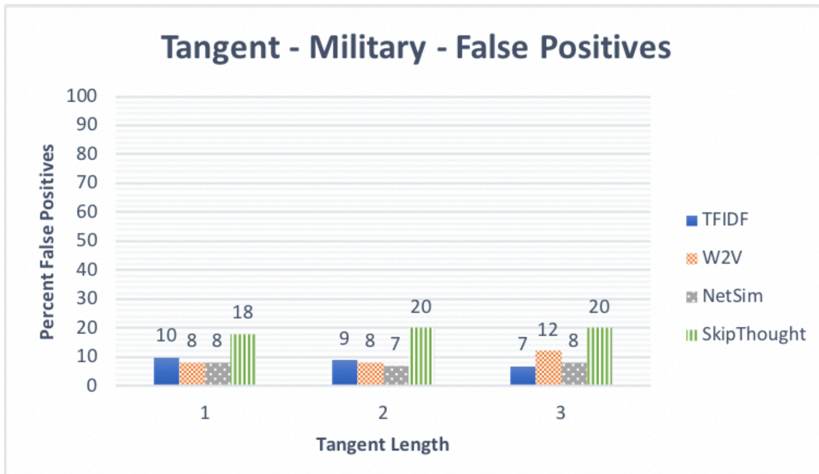

Figure 17. The false positive rate for TRAP in the military domain.

In the concierge domain, 19 dialogues were used at three separate tangent lengths for a total of 57 TRAP trials each using Word2Vec, NetSim, TFIDF and SkipThoughts. A grand total of 228 TRAP trials were performed in the concierge domain. Fig. 18 shows that TRAP with Word2Vec again performed better than TRAP with TFIDF, NetSim or SkipThoughts with all generally worsening as the tangent length grew. The exception to this again was TRAP with NetSim which followed a similar arch in performance as it did in the military domain.

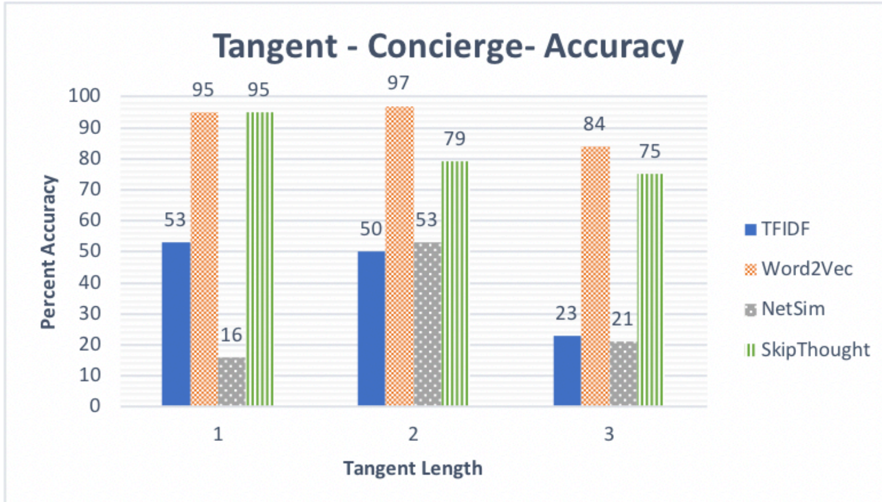

Figure 18. The accuracy of TRAP in the concierge domain.

The false positive rates for TFIDF generally fell as the tangent length grew see Fig. 19. This could be because as the tangent length grew, TFIDF had fewer cosine scores above the threshold thereby making catching fewer tangents but also flagging fewer non-tangential questions. When TRAP used Word2Vec, the false positive rate grew with its highest score being 6% at a tangent length of three. TRAP with NetSim performed significantly worse in the concierge domain with the false positive rate being four times higher than in the military domain on a tangent length of two. TRAP with NetSim also had the highest false positive rates on tangent sizes one and two but fell below TRAP with SkipThoughts on a tangent of size 3.

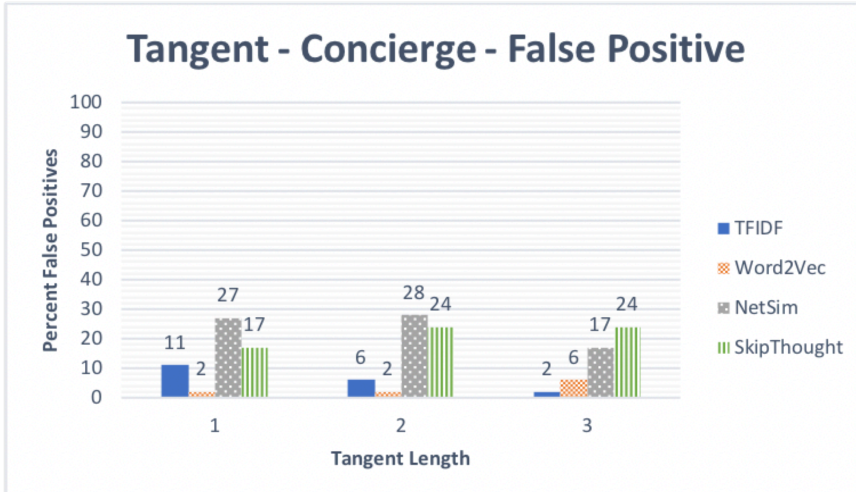

Figure 19. The false positive rate of TRAP in the concierge domain.

10. Related Research

Our approach builds upon previous work, particularly a taxonomy of goals (Bengfort & Cox, 2015) to create a multidimensional representation of a knowledge goal. This representation is defined by a knowledge goal space with which we can compare goal similarity using distance metrics. This implementation therefore allows us to use a simple nearest neighbor algorithm to provide guidance to the user; a simplification that improves upon many challenges regarding case-based learning.

When information is shared between team members on a project, their text is usually within a domain or along the lines of a specific topic of pursuit. The work done by (Hwang et. al, 2012) presents a context awareness system that learns about the user's needs and understands the domain the problem has appeared within. By collecting other data about their users, these systems can establish a context to better tailor their results and information to the user. They use a technology called ubiquitous learning. With this technology, alongside k nearest neighbors their system understands the appropriate context for learners in a learning environment.

The work of (Aha et. al, 2005) is highly relevant due to the conversation-based interface. Their work highlighted the importance of refining the user's question to solve a problem.

Such problems faced by the user are usually vaguely or briefly defined and lack adequate detail for the system to provide a meaningful solution. Their method of implementing a conversational style interface to extracting details of a target goal was proven to be effective and close to the natural way humans communicate problems. Our work builds on this style of conversation-based interface and also tracks the user's decomposition of goals into sub-goals. By allowing users to map out a 'plan' to solve their target goal, our system can better understand the context of why goals change and identify false tangents to better provide guidance.

The work done in (Higgins et. al, 2006) is relevant as the goal of identifying off topic answers to questions with TRAP. Their work focuses on content vector analysis (CVA) and unlike TRAP which uses cosine similarity and a hyperspace for tangent recognition, CVA uses a variant of the inverse document frequency score and cosine similarity to measure the relationship between the answer and the question. While both approaches use cosine similarity, TRAP uses a different vector representation that is determined by the surrounding questions in a dialogue.

In doing so, the number of dimensions grows as the number of questions in the dialogue increases. CVA uses a vector representation that is based on the content of the question and the answer supplied by the student. Furthermore, the vector representation generated by TRAP is dependent on the similarity measure used by TRAP. CVA as described by the paper, only uses a variant of TFIDF. While TFIDF can be used with TRAP, so too can Word2Vec, NetSim and Skip-Thoughts. The decoupling of the similarity measure and the algorithm make TRAP more versatile and less susceptible to shortcomings of any one similarity measure.

The work done in (Stewart et. al, 2006) is also relevant as the dialogue nature of telephone speech is similar to the dialogue interface in Ronin. Their work focuses on a machine learning algorithmic approach aimed at automating the identification of irrelevance within dialogues. In doing so, they've built a classifier that identifies important features of dialogues. Since their approach incorporates a series of text over a period of time, this shares similarities to Ronin's goal trajectory data structure. Yet Ronin's approach differs in that Ronin neither uses a classifier or any other machine learning technique for detecting off-topic text.

11. Discussion.

The work focuses on representing knowledge goals and measuring the similarity between them. Case-based reasoning systems are suitable for domains where knowledge goals are similar across many users. The interaction between users and case-based reasoning systems can capture increasingly complex knowledge investigations and retain them as goal trajectories within their case-base. In order to improve retrieval of cases for these systems, it is important to consider additional conceptual, contextual and task related information. Our method for retrieval demonstrates an effective way to retrieve goal trajectory cases for knowledge investigations across four different ways of measuring similarity. Because of the flexibility of our system, it can be applied to many different domains so long as knowledge goals within those domains can be decomposed into their most basic conceptual, contextual and task components.

The issue of capturing a tangent where the anomaly is the first question in the dialogue has been a challenge for our evaluation. For this reason, we adopted the simple assumption that the first question was not a tangent. However, while this assumption allowed us to proceed with an initial evaluation to gauge the relevance and utility of TRAP, it is not robust or realistic. For this reason, we are continuing to explore options for adding context to determine the substance of the first question. One such aforementioned approach is what we refer to as idealization. This approach extracts the highest non-perfect score from each column in the dialogue to create an ideal vector. In a sense this would be a question that is highly related to all other questions in the dialogue if such a thing existed and is therefore an ideal. By inserting this vector before the first question vector in the dialogue, we provide initial context in which to evaluate the first question. While this approach remains to be evaluated the underlying challenge does underscore the non-triviality of capturing a tangent whose anomaly is the first question.

The degree to which a case-based reasoning system can assist a user in finding solutions to novel problems largely rests on the system's capacity to retain, retrieve, revise and reuse its experiences. Yet those experiences which the system has acquired through interactions with past users often contain the mistakes their users also made. To maximize the positive impact a system has while minimizing its negative affects requires a system to both recognize past mistakes and prevent those mistakes from being repeated. Indeed, a system which accumulates cases that routinely commit the same mistakes is a system that can be improved to say the least.

By modeling cases in the form of trajectories through a multi- dimensional space, these mistakes can manifest in diverging tangents. Therefore, combining both tangent recognition and pruning of the anomalies which created them leads to an overall better experience for users. TRAP combined with the Word2Vec neural network has been shown to be effective at removing tangents from natural language dialogues while having a low false positive rate. These features make TRAP suitable for processing error from cases to be efficiently reused to find future solutions.

11.1. Future Research.

The thesis work has broad applications and can be improved with the following research:

- The research work can be improved by increasing the number of cases in the case-base for both domains. Increasing the number of cases would enable more robust evaluations.
- Finding a way to address the assumption that the first question cannot be a tangent will improve the overall performance of TRAP. One method discussed was idealization, but no evaluation was performed to test whether this is an effective way to detect the tangents that appear as the first question.
- Adding more features to user profiles in addition to age, marital status and gender will allow for a better evaluation of contextual similarity and could be useful to TRAP.
- While TRAP has demonstrated a strong ability to recognize hard tangents, no evaluation has been performed on TRAP to recognize soft tangents which are more difficult to detect and arguably more likely to occur.
- Adaptation of cases also has not been implemented. While adaptation remains a persistent challenge to case-based reasoning in general, Ronin's performance can be further improved with the compliment of the capacity to adapt cases to novel problems.
- Retrieval only considers pairwise questions when measuring concept similarity but considering nearby questions can make retrieval more robust and less susceptible

to dialogues where the best matching question between the current dialogue and the case dialogue are offset.

11.2. Conclusion.

As artificial intelligence becomes a more prevalent technology the ability to understand natural language will become increasingly sophisticated. Much like visual components have been the standard for interfacing with computers, it may well be the case that natural language will be the user's interface of choice for artificially intelligent systems. Case-based reasoning likely will continue to become more relevant as it can be implemented in a wide number of applications without the need for large training datasets.

Appendix A.

Dialogue 1

[Are transport facilities available here?, Are there any rental car stations nearby?, Where can I the nearest airport?, Does this hotel provide any transport facilities?, Are any public transport facilities available here?]

Dialogue 2

[What are the tourist attractions in this place?, What type of tourist places are available here?, Where can we go to shopping in this place?, Where would I find parks situated close to me?, Where can I find the nearest beach?, Are there any theatres located nearby?]

Dialogue 3

[Where would I be able to get delicious food in this city?, Are there any buffets available nearby?, Can we find any Italian food here?, Is Mexican food tasty here?, Are there any mixed cuisine restaurants available here?]

Dialogue 4

[Can I know the weather forecast today?, Can we go out for visiting this place in this weather?, Is it suggestible to leave this place today in this weather?, Should we wear winter clothing to go out?, In which season is it suggestible to visit this place?]

Dialogue 5

[Where can I find ATM machines nearby?, Are there any international banks available here?, Do your hotel provide any bank facilities?, Can any representative from your hotel could assist me to the bank?, What is the money conversion rate?]

Dialogue 6

[Where can I find the nearest hospital?, Can I find any respiratory therapist hospitals near me?, Where can I find a dental hospital here?, Are there any emergency hospitals nearby?, Where can I find children's hospital nearby?, Are there any event organizers available here?, Are there any bouquet delivery shops located here?, Where can I find gift shops nearby?, What kind of souvenirs are unique to this place?, Where can we have a surprise party here?]

Dialogue 7

[Where can I find a super market here?, Where can I find laundry services here?, What WIFI or internet services do you offer?, Does the room have a kitchen?, How often is room service done?]

Dialogue 8

[What is the real estate market like?, How expensive is it to invest in housing here?, Are there any houses for sale here?, Can I find any land for sale here?, What returns can we expect from local real estate?]

Dialogue 9

[Where can I get for dinner tonight?, How expensive are local restaurants?, Are there any bars have good food?, Which nearby restaurants also serve alcohol?, Which restaurants also have happy hours?]

Dialogue 10

[What are some fun outdoor activities?, Are there any beautiful natural places or parks we could visit?, Is there an animal refuge or wild life sanctuary we can tour?, Can we camp at the wild life sanctuary?, Can we go rock climbing at the wild life sanctuary?, What considerations should we make when we go camping?]

Dialogue 11

[What kind of assistance and or accommodations does the hotel provide for disabled guests?, Does the hotel have easy wheelchair access?, Does the hotel offer wheelchair rental?, Would it be possible to reserve a room on the hotel's first floor?, What tourist sights are there that are also wheelchair accessible?, What kind of assistance and or accommodations does the hotel provide for disabled guests?, what are some accommodating modes of transportation when we go sightseeing downtown?]

Dialogue 12

[Are there any good food trucks in Dayton?, How can I locate a nearby food truck?, Are food trucks generally that clean?, Are there any movies about food trucks?, What is the history of food truck cuisine?]

Dialogue 13

[How safe is it to go out at night?, Are there any police stations nearby?, How prevalent is crime in neighborhoods?, What kind of scams or fraud should we be aware of?, How can we contact the police?]

Dialogue 14

[What traditions are practiced here?, Are there any festivals or parades we can join?, What holidays are there this month?, Are there currently any historical celebrations?, What are the main tribal communities here?]

Dialogue 15

[What beaches or water parks are nearby?, Can we get food near the beach?, How popular are beaches this time of year?, How cold is the water this season?, Do the beaches have lifeguards?]

Dialogue 16

[How close is the nearest theme park?, How expensive are tickets to the theme park?, What kind of rides are at the theme park?, Is it expensive to order food while at the theme park?, How expensive are theme park tickets?, Can we purchase tickets online?]

Dialogue 17

[Can we go gambling somewhere?, What casinos are high end?, Can we play poker?, What are the odds of winning roulette?, Where is the nearest casino?]

Dialogue 18

[Can you suggest a spa we could visit?, Are there tanning beds at the spa?, Does the spa do manicures?, Where can we get the best massages?, Is there a hair salon nearby?]

Dialogue 19

[Do we find any transport facilities here?, Can I find the nearest car rental station?, Where can I find airport nearby from this place?, Can we expect any transport facilities from this hotel?, Do we find public transport in this place?]

References.

Aamodt A., & Plaza E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*. 7(1), 39-59.

Aha, D., Breslow, L., & Muoz-Avila, H. (1999). Conversational case-based reasoning. *Applied Intelligence*. 1-25.

Aha, D., McSherry, D., & Yang, Q. (2005). Advances in conversational case-based reasoning. *The Knowledge Engineering Review*, 20(3), 247–254.

Bengfort, B., & Cox, M. T. (2015). Interactive reasoning to solve knowledge goals. In D. W. Aha (Ed.), *Goal reasoning: Papers from the ACS workshop* (pp. 10-25). Tech. Rep. No. GT-IRIM-CR-2015-001. Atlanta, GA: Georgia Institute of Technology, Institute for Robotics and Intelligent Machines.

Branting K., Lester J., & Mott B. (2004). Dialogue Management for Conversational Case-Based Reasoning. *European Conference for Case-based Reasoning,* 3155.

Cox, M. T., Alavi, Z., Dannenhauer, D., Eyorokon, V., Munoz-Avila, H., & Perlis, D. (2016). MIDCA: A metacognitive, integrated dual cycle architecture for self regulated autonomy. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence.* 5, 3712-3718. Palo Alto, CA: AAAI Press.

Cox, M. T., Dannenhauer, D., & Kondrakunta, S. (2017). Goal operations for cognitive systems. *In Proceedings of the Thirty-first AAAI Conference on Artificial Intelligence.* 4385-4391. Palo Alto, CA: AAAI Press.

Cox, M. T., & Veloso, M. M. (1998). Goal transformations in continuous planning. In M. desJardins (Ed.), *Proceedings of the 1998 AAAI Fall Symposium on Distributed Continual Planning*. 23-30. Menlo Park, CA: AAAI Press / The MIT Press.

de Mantaras, R.L., McSherry, D., Bridge, D., Leke, D., Smyth, B., Craw, S., Faltings B., Maher, M.L., Cox, M.T., Forbus K., Keane, M., Aamodt, A., & Watson, I. (2015). Retrieval, reuse, revision and retention in case-based reasoning. *The Knowledge Engineering Review*.

Eyorokon, V., Bengfort, B., Panjala, U., & Cox, M. T. (2016). Case-Based Goal Trajectories for Knowledge Investigations. In A. Coman & S. Kapetanakis (Eds.), *Twenty-Forth International Conference on Case-Based Reasoning Workshop Proceedings: Synergies between CBR and Knowledge Discovery*. 202-211. Atlanta, GA.

Eyorokon, V., Gogineni, V., Yalamanchili, P., & Cox, M. T. (in press). Case retrieval using goal similarity for knowledge investigations. To appear in *Proceedings of the Sixth Annual Conference on Advances in Cognitive System*s. Palo Alto, CA: Cognitive Systems Foundation.

Eyorokon, V., Panjala, U., & Cox, M. T. (2017). Case-based goal trajectories for knowledge investigations. In V. Rus & Z. Markov (Eds.), *Proceedings of the 30th International FLAIRS Conference* (pp. 477-482). Palo Alto, CA: AAAI Press.

Eyorokon, V., Yalamanchili, P., & Cox, M. T. (2018). Tangent recognition and anomaly prun-ing to trap off-topic questions in conversational case-based dialogues. In M. Cox, P. Funk, & S. Begum (Eds.), *Case-Based Reasoning Research and*

Development: Proceedings of the 26th International Conference (pp.213-228). Berlin: Springer.

Gu, M., & Aamodt, A. (2006). Dialog Learning in Conversational CBR. *Florida Artificial Intelligence Research Society Conference*.

Harris, Z. S. (1954). Distributional structure. *Word, 10,* 146-162.

Higgins, D., Burstein, J., & Attali, Y. (2006). Identifying off-topic student essays without topic-specific training data. *Natural Language Engineering*. 12, 145-159.

Huang, A., Milne, D.N., Frank, E., & Witten, I.H. (2009). Clustering Documents Using a Wikipedia-Based Concept Representation. *Pacific-Asia Conference on Knowledge Discovery and Data Mining*.

Hwang, G., Tsai, C., Chu, H., Kinshuk K., & Chen, C. (2012). A context-aware ubiquitous learning approach to conducting scientific inquiry activities in a science park. *Australasian Journal of Educational Technology*, 28, 931-947.

Kapetanakis (Eds.) (2016). Synergies between CBR and Knowledge Discovery. *Twenty-Forth International Conference on Case-Based Reasoning Workshop Proceeding.*

Kiros, R., Zhu, Y., Salakhutdinov, R., Zemel, R.S., Urtasun, R., Torralba, A., & Fidler, S. (2015). Skip-Thought Vectors. *Neural Information Processing Systems.*

Kolodner, J. L. (1993). Case-based reasoning. San Mateo, CA: *Morgan Kaufmann Publishers*.

Kryszkiewicz M. (2014). The Cosine Similarity in Terms of the Euclidean Distance. *Encyclopedia of Business Analytics and Optimization*. 4, 223.

Leake, D.B. (1996). CBR in Context: The Present and Future. *AAAI.* 1-30.

Li, Y., McLean, D., Bandar, Z., O'Shea, J., & Crockett, K.A. (2006). Sentence similarity based on semantic nets and corpus statistics. *IEEE Transactions on Knowledge and Data Engineering,* 18, 1138-1150.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. *NIPS*.

Panjala, U., Eyorokon, V., & Cox, M. T. (2017). A mixed-initiative approach for knowledge discovery. In A. R. Bryant, J. R. Lopez, Jr., & R. F. Mills (Eds.), *Proceedings of the 12th International Conference on Cyber Warfare and Security - ICCWS 2017* (pp. 533-536). Sonning Common, UK: Academic Conferences and Publishing International.

Ram, A. (1991). A theory of questions and question asking. *Journal of the Learning Sciences*, 1, 273-318.

Ram, A., & Hunter, L. (1992). The use of explicit goals for knowledge to guide inference and learning. *Applied Intelligence, 2*, 47-73.

Recio-Garćıa, J.A., Díaz-Agudo, B., & González-Calero, P.A. (2007). Textual CBR in jCOLIBRI : From Retrieval to Reuse?

Riesbeck, C., Schank, R., & Riesbeck, C. (Ed.) (1989). Inside case-based reasoning. *Lawrence Erlbaum Associates*.

Salton, G., Wong, A., & Yang, C. (1975). A vector space model for automatic indexing. *Common ACM, 18*, 613-620.

Schumacher, P., Minor, M., Walter, K., & Bergmann, R. (2012). Extraction of procedural knowledge from the web: a comparison of two workflow extraction approaches. *WWW*.

Singhal, A. (2001). Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.,* 24, 35-43.

Stewart, R., Danyluk, A., & Liu, Y. (2006). Off-topic detection in conversational telephone speech. *Proceedings of the HLT-NAACL 2006 Workshop on Analyzing Conversations in Text and Speech*, 8-14.

Weber, R., Martins, A., Barcia, R.M. (1998). On legal texts and cases. In: *Textual case-based reasoning: Papers from the AAAI 1998 Workshop (Technical Report WS-98-12)*, 40-55. AAAI Press, Menlo Park.

Weber, R.O., Ashley, K.D., & Brüninghaus, S. (2005). Textual case-based reasoning. *Knowledge Eng. Review,* 20, 255-260.